VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Rozšíření programu UniSave o možnost načítání dat z multimetru METEX M3660D

UniSave Extension for Device METEX M3660D

2011 Matěj Děcký

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 16. dubna 2009 .....

Rád bych na tomto místě poděkovala všem, kteří mi s prací pomohli, protože bez nich by tato práce nevznikla.

Abstrakt

Základem této bakalářské práce je rozšíření aplikace Unisave o načítání z multimetru značky METEX, které se využivá u elektrických měření. Tato práce obsahuje základní analýzu jak aplikace Unisave, tak i programu CommBridge, který se stará o samotné spojení s přístrojem a komunikaci mezi aplikací Unisave a multimetrem. Dále je také popsán obecný postup pro přidávání nových typů měření a nových měřidel do aplikace.

Klíčová slova: java, C++, Unisave

Abstract

Basic of this bachelor work is extension of Unisave aplication by METEX multimeter which is used for measuring of electrical units. This work include basic analysis for Unisave aplication and the CommBridge program which is used for connection with multimeter and communication between Unisave aplication and multimeter. Next thing that is described is universal method for adding new types of mesurment and mesured devices into aplication.

Keywords: java, C++, Unisave

Seznam použitých zkratek a symbolů

GUI	–	Graphical user interface
IDE	–	Integrated development environment
SW	–	Software
HW	–	Hardware
UML	–	Unified Modeling Language
XML	–	Extensible Markup Language

Obsah

Seznam tabulek

Seznam obrázků

Seznam výpisů zdrojového kódu

1 Úvod

Kvůli propojení vnějších zařízení s počítačem bylo potřeba, aby osobní počítače byly vybaveny rozhraním, které by s takovýmito zařízeními mohly komunikovat. V době největšího rozvoje se tímto rozhraním stal sériový COM port. S pomocí tohoto portu se přenášela data jak mezi přímými periferiemi (počítačová myš) tak s nepřímými. Takovýmto nepřímým zařízením jsou i digitální multimetry, který představuje například METEX M3660D.

Tyto multimetry jsou dodávány spolu se SW, který dokázal naměřené data reprezentovat, avšak vzhledem k vývoji operačních systémů a programovacích jazyků se tyto programy staly dále nepoužitelné.

Cílem mojí bakalářské práce je implementovat načítání z multimetru METEX M3660D do programu Unisave a vytvořit tak návod pro možnou implementaci jiných multimetrů. Práce na implementaci bude probíhat v následujících krocích:

- Analýza komunikace multimetru s dodávaným SW
- Analýza programu Commbridge, možnosti způsobu komunikace s multimetrem a výsledná implementace
- Úprava programu Unisave

2 Analýza komunikace multimetru s dodávaným SW

2.1 Popis komunikačního rozhraní

Jak již bylo řešeno v úvodu, multimetry komunikují s osobním počítačem přes rozhraní sériového portu COM. Na HW úrovní tyto zařízení komunikují přes standart RS-232. Klasicky se v osobnímu počítači připojuje přes devíti pinový D-Sub konektor DE-9, avšak vzhledem k ústupu tohoto konektoru a vytlačení modernějším USB se požívá převážně už jen v průmyslové sféře. Pro použití v oblasti moderních PC je použito převodníků z USB na konektor DE-9, využívající sběrnici RS-232[1]. Tyto převodníky jsou převážně tvořené čipy FT232, což má ovšem za následek zvýšení zpoždění a následnou možnou nekompatibilitu se softwarem.

RS-232 používá pro svou komunikaci dvě úrovně, logickou 0 a 1. Logická 1 je někdy označována jako marking state nebo také klidový stav a logická 0 jako space state. Napěťové úrovně jsou znázorněny v tabulce 1 a ukázka průběhu jak je viděna na osciloskopu je ukázána na obrázku 1.

2.1.1 Analýza komunikace

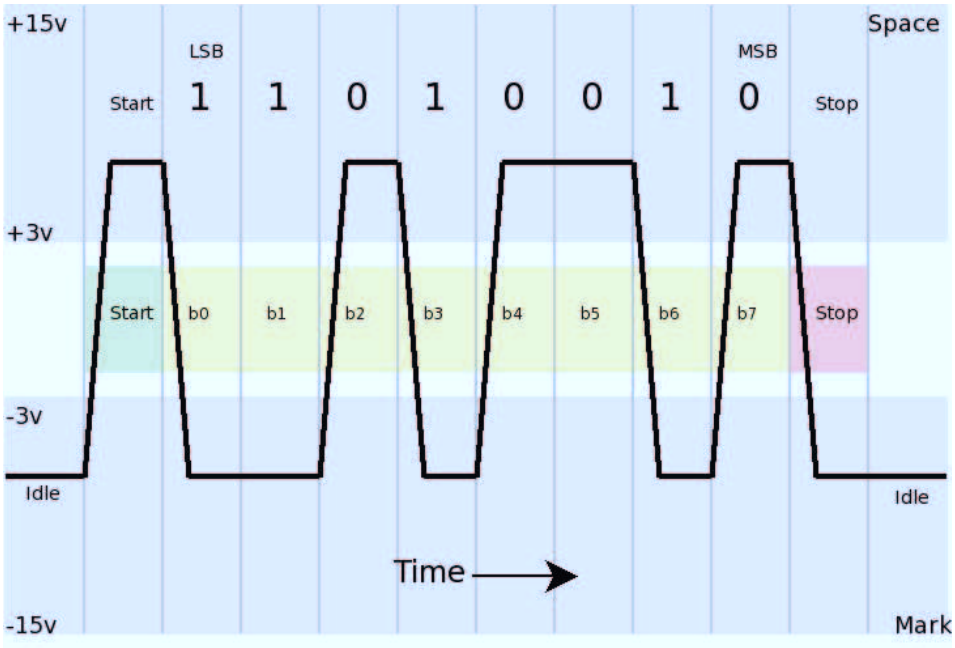
Jelikož zdrojové kódy dodávané aplikace nejsou k dostání, bylo třeba zjistit, jak přesně dodané multimetry komunikují. K tomu bylo potřeba zachytit komunikaci na COM portu pomocí aplikace Portmon [5], která dokáže zachytit aktivitu na všech sériových i paralelních portech. Tato aplikace je volně dostupná(freeware) přes webové rozhraní Windows sysinternals.

Po připojení měřidla a spuštění načítání hodnot z programu dodávaném výrobcem jsem odchytil následující komunikaci:

```
0  16:55:19  ntvdm.exe IRP_MJ_CREATE Serial2 Options: Open
0  16:55:19  SUCCESS
1  16:55:19  ntvdm.exe IOCTL_SERIAL_SET_QUEUE_SIZE Serial2 InSize: 2048 OutSize: 2048
1  16:55:19  SUCCESS
2  16:55:19  ntvdm.exe IOCTL_SERIAL_SET_TIMEOUTS Serial2 RI:−1 RM:0 RC:0 WM:0 WC
     :65000
19  16:55:19  ntvdm.exe IOCTL_SERIAL_SET_BAUD_RATE Serial2 Rate: 1200
19  16:55:19  SUCCESS
20  16:55:20  ntvdm.exe IOCTL_SERIAL_CLR_RTS Serial2
20  16:55:20  SUCCESS
21  16:55:20  ntvdm.exe IOCTL_SERIAL_SET_DTR Serial2
21  16:55:20  SUCCESS
22  16:55:20  ntvdm.exe IOCTL_SERIAL_SET_LINE_CONTROL Serial2 StopBits: 1 Parity: NONE
     WordLength: 7
22  16:55:20  SUCCESS
23  16:55:20  ntvdm.exe IOCTL_SERIAL_SET_CHAR Serial2 EOF:0 ERR:0 BRK:0 EVT:0 XON:1
     XOFF:13
23  16:55:20  SUCCESS
24  16:55:20  ntvdm.exe IOCTL_SERIAL_SET_HANDFLOW Serial2 Shake:1 Replace:0 XonLimit:0
     XoffLimit:1536
24  16:55:20  SUCCESS
25  16:55:20  ntvdm.exe IOCTL_SERIAL_GET_MODEMSTATUS Serial2
```

Úroveň	Vysílač	Příjmač
Log. 0	+5V až +15V	+3V až +25V
Log. 1	-5V až -15V	-3V až -25V
Nedefinováno	-3V až +3V	

Tabulka 1: Tabulka napěťový úrovní standartu RS-232



Obrázek 1: Průběh signálu RS-232

```
25  16:55:20  SUCCESS
26  16:55:20  ntvdm.exe IOCTL_SERIAL_SET_WAIT_MASK Serial2 Mask: RXCHAR RXFLAG
     TXEMPTY CTS DSR RLSD BRK ERR RING
26  16:55:20  SUCCESS
27  16:55:20  ntvdm.exe IOCTL_SERIAL_GET_MODEMSTATUS Serial2
27  16:55:20  SUCCESS
28  16:55:20  ntvdm.exe IOCTL_SERIAL_WAIT_ON_MASK Serial2
29  16:55:20  ntvdm.exe IRP_MJ_WRITE Serial2 Length 2: D.
60  16:55:21  ntvdm.exe IOCTL_SERIAL_GET_MODEMSTATUS Serial2
60  16:55:21  SUCCESS
61  16:55:21  ntvdm.exe IOCTL_SERIAL_WAIT_ON_MASK Serial2
61  16:55:21  SUCCESS
62  16:55:21  ntvdm.exe IOCTL_SERIAL_GET_MODEMSTATUS Serial2
62  16:55:21  SUCCESS
63  16:55:21  ntvdm.exe IOCTL_SERIAL_WAIT_ON_MASK Serial2
64  16:55:21  ntvdm.exe IRP_MJ_READ Serial2 Length 100
64  16:55:21  SUCCESS Length 14: DC −0.000 V.
65  16:55:21  ntvdm.exe IRP_MJ_WRITE Serial2 Length 2: D.
```

Výpis 1: Výpis komunikace z multimetru METEX

Jak je vidět na výpisu 1, tak nejprve se nastaví parametry, které jsou vysvětleny v kapitole 3.2. Po nastavení těchto parametrů (řádky 2-24) se nastaví čekací maska (řádek 26), která se následně neustále testuje na příznak zápisu. Před každým vyčtením hodnoty se musí do měřidla poslat libovolný znak(v tomto případě znak D) a poté se čeká na odpověď. Přijatá odpověď má formát názevVeličinySPCnaměřenáHodnotaSPCjednotkaCR.

Při analýze druhého zapůjčeného multimetru od firmy Protek jsem zjistil, že jeho komunikace se svým SW je, až na lehce odlišné parametry nastavení, naprosto totožná s multimetrem METEX. Z toho jsem usoudil, že tento způsob komunikace bude implementován u většiny měřidel.

3 Program Commbridge

Commbridge je program napsaný v jazyce C++, který je součástí aplikace Unisave jako externí program. Tato možnost je možná lehce zbytečná, jelikož jazyk java umožňuje přístup k sériovému portu, ale zřejme byla autorem vyhodnocena jako efektivnější. Jak již bylo řečeno, program je součástí aplikace Unisave, která ho spouští a nastaví společný standartní vstup a výstup. Přes tyto vstupy a výstupy je pak prováděna veškerá kominikace jako informace o typu zařízení, jejich nastavovací údaje čtené z XML souboru a zpětné posílání přijatých dat z měřidla.

3.1 Komunikace se sériovým portem v jazyce C++

První věc, kterou je třeba si uvědomit, je jak probíhá komunikace a jaká knihovna a v ní uložené metody jsou potřeba pro úspěšnou komunikaci ze sériového portu. V jazyce C++ jsou metody pro práci se IO zařízeními uloženy v základní knihovně windows.h. Obecně komunikace probíhá takto:

- Otevře se "soubor"(sériový port), pomocí funkce CreateFile.
- Nastaví se parametry sériového portu pomocí funkce SetCommState. Zde se využívá DCB struktura[2] popsána v kapitole 3.2
- Nastaví se parametry časovačů sériového portu pomocí funkce SetCommTimeouts. Zde se využívá struktura commTimeouts [3].
- Následně se může buď číst či zapisovat na port pomocí funkcí ReadFile a WriteFile.
- Uzavře se port funkcí CloseHandle.

3.2 Nastavení sériového portu

Pro nastavení parametrů se používá stuktura DCB struktura, jejíž obsah je znázorněn na výpisu 2. I když všechny tyto parametry musí být nastaveny pro funkční komunikaci, u velké části z nich si vystačíme s defaultní hodnotou. Parametry které se nastavovat musí jsou rychlost přenosu, parita, počet stop bitů a parametry ovládající provoz přenosu. Přesné názvy parametrů a jejich možnosti konfigurace jsou:

- BaudRate - rychlost přenostu dat
- ByteSize - počet vysílaných a přijímaných bitů v bytu, ve výpisu odchycené komunikace je reprezentován parametrem WordLength
- fDtrControl - nastavuje řídící signál DTR, který může nabývat hodnot:
  - DTR_CONTROL_DISABLE(0x00)
  - DTR_CONTROL_ENABLE(0x01)
  - DTR_CONTROL_HANDSHAKE(0x02)

Ve výpisu je reprezentován řídícím signálem IOCTL_SERIAL_SET_DTR

- fRtsControl - nastavuje řídící signál RTS, který může nabývat hodnot:
  - RTS_CONTROL_DISABLE(0x00)
  - RTS_CONTROL_ENABLE(0x01)
  - RTS_CONTROL_HANDSHAKE(0x02)
  - a RTS_CONTROL_TOGGLE(0x03)

  Ve výpisu je reprezentován řídícím signálem IOCTL_SERIAL_CLR_RTS
- fOutxCtsFlow a fOutxDsrFlow - signály monitorující cts a dts. Nastavujeme na hodnotu TRUE pouze pokud parametry fDtrControl a fRtsControl nabývají hodnoty 0x02. V opačném případě nastavujeme FALSE.
- Parity - určuje paritní bit přenosu. Nabývá hodnot EVENPARITY, MARKPARITY, NOPARITY, ODDPARITY nebo SPACEPARITY. Ve výpisu je reprezentován parametrem se stejným názvem.
- StopBits - určuje počet stop bitů použitých v přenosu. Hodnoty jsou ONESTOPBIT, ONE5STOPBITS a TWOSTOPBITS.
- XOffChar a XOnChar - hodnota znaků XON a XOFF použitých jak v u vysílání tak u přijímání.
- XOffLim - minimální počet uskladněných volných bytů ve vstupním bufferu před aktivováním kontroly toku (flow control). Tato hodnota by nikdy neměla být nulová.
- XOnLim - minimální počet užívaných bytů ve vstupním bufferu před aktivováním kontroly toku (flow control).

Další parametry, které je třeba nastavit jsou časovače sériového portu commtimeouts. Struktura je vypsána na výpisu 3. Parametry zde nastavené ovlivňují chování funkcí ReadFile, WriteFile, ReadFileEx, a WriteFileEx.

- ReadIntervalTimeout - Maximální doba v milisekundách, která je umožněna mezi příchodem dvou bytů na komunikační lince. Během operace ReadFile začíná tento časový úsek přijetím prvního bytu. Pokud interval mezi příjetím dvou bytů přeročí tuto hodnotu, operace ReadFile bude ukončena a vrátí všechny doposud přijaté data. Hodnota nula pro tento parametr značí, že není použit.
- ReadTotalTimeoutMultiplier - Násobička, použitá pro vypočítání celkové periody u všech operací čtení, v milisekundách. Pro každou tuto operaci je tato hodnota vynásobena požadovaným počtem bytů k přečtení.

- ReadTotalTimeoutConstant - Konstanta, použitá pro vypočítání celkové periody u všech operací čtení, v milisekundách. Pro každou tuto operaci je tato hodnota připočtena k parametru ReadTotalTimeoutMultiplier a požadovanému počtu bytů. Hodnota nula pro oba parametry ReadTotalTimeoutMultiplier a ReadTotalTimeoutConstant značí, že nebudou použity pro jakoukoliv operaci čtení.
- WriteTotalTimeoutMultiplier - Násobička, použitá pro vypočítání celkové periody u všech operací zápisu, v milisekundách. Pro každou tuto operaci je tato hodnota vynásobena požadovaným počtem bytů k zápsání.
- WriteTotalTimeoutConstant - Konstanta, použitá pro vypočítání celkové periody u všech operací zápisu, v milisekundách. Pro každou tuto operaci je tato hodnota připočtena k parametru WriteTotalTimeoutMultiplier a požadovanému počtu bytů. Hodnota nula pro oba parametry WriteTotalTimeoutMultiplier a WriteTotalTimeoutConstant značí, že nebudou použity pro jakoukoliv operaci zápisu.

Při testování aplikace a její spojení se zapůjčenými měřidly bylo zjištěno, že nejlépe komunikují s nastavení ve výpisu 4. Jeliž se hodnoty těchto parametrů posílají externě, nemůžeme použít makro MAXWORD, ale nastavíme parametr místo tohoto makra na hodnotu -1.

```
typedef struct _DCB {
  DWORD DCBlength;
  DWORD BaudRate;
  DWORD fBinary :1;
  DWORD fParity :1;
  DWORD fOutxCtsFlow :1;
  DWORD fOutxDsrFlow :1;
  DWORD fDtrControl :2;
  DWORD fDsrSensitivity :1;
  DWORD fTXContinueOnXoff :1;
  DWORD fOutX :1;
  DWORD fInX :1;
  DWORD fErrorChar :1;
  DWORD fNull :1;
  DWORD fRtsControl :2;
  DWORD fAbortOnError :1;
  DWORD fDummy2 :17;
  WORD wReserved;
  WORD XonLim;
  WORD XoffLim;
  BYTE ByteSize;
  BYTE Parity;
  BYTE StopBits;
  char XonChar;
  char XoffChar;
  char ErrorChar;
  char EofChar;
  char EvtChar;
  WORD wReserved1;
} DCB, *LPDCB;
```

Výpis 2: DCB struktura

```
typedef struct _COMMTIMEOUTS {
  DWORD ReadIntervalTimeout;
  DWORD ReadTotalTimeoutMultiplier;
  DWORD ReadTotalTimeoutConstant;
  DWORD WriteTotalTimeoutMultiplier;
  DWORD WriteTotalTimeoutConstant;
} COMMTIMEOUTS, *LPCOMMTIMEOUTS;
```

Výpis 3: COMMTIMEOUTS struktura

```
comTimeouts.ReadIntervalTimeout = MAXDWORD;
comTimeouts.ReadTotalTimeoutConstant = 0;
comTimeouts.ReadTotalTimeoutMultiplier = 0;
comTimeouts.WriteTotalTimeoutConstant = 0;
comTimeouts.WriteTotalTimeoutMultiplier = 0;
```

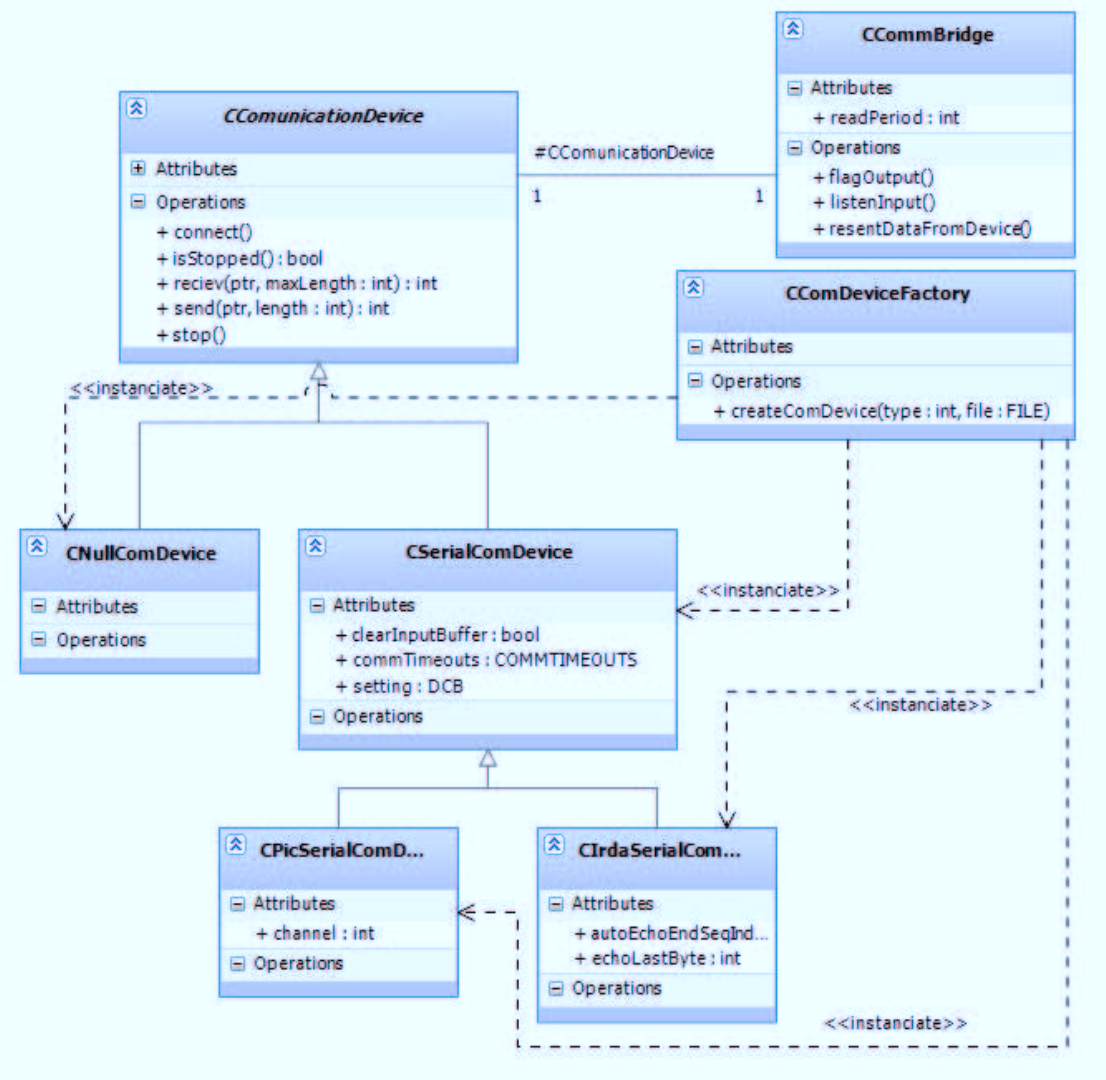
Výpis 4: Nastavení commtimeouts pro měřidlo METEX

3.3 Analýza funkčnosti programu

Program commbridge je prakticky složen ze dvou částí. První částí jsou třídy CommunicationDevice a ComDeviceFactory a třídy jim podřízené. Třída CommunicationDevice je abstraktní třídou(pouze definuje metody, které však neimplementuje), ze které dědí všechy třídy reprezenující konkrétní zařízení jako je například třída SerialComDevice. Tyto třídy v sobě již mají metody implementovány. Třída ComDeviceFactory implementuje jedinou metodu a tou je createComDevice, sloužící k vytvoření požadovaného typu zařízení, dodaného externě jako celočíselný parametr. Druhou částí programu je třída CommBridge. Ta implementuje mezi jinými 4 nejdůležitější metody. Jsou jimi run, listenInput, flagOutput a representDataFromDevice. V metodě run se načítá perioda čtení a hlavně typ zařízení s nimž budeme pracovat.Třídní diagram programu je zobrazen na obrázku 2. Celá komunikace pracuje ve třech oddělených vláknech:

- Prvním vláknem je vláknno, které je spuštěno jako metoda listenInput. Ta slouží pro načítání standartního vstupu, tedy jako komunikaci s aplikací Unisave. V současné době je implementováno pouze ukončení načítání z měřidla a to buď přijetím EOF znaku nebo přijetím řetězce "KONEC".

- Druhé vlákno je spuštění metody flagOutput, která neustále posílá znak 'M' na zařízení, ze kterého vyčítáme data. Jelikož multimetr METEX posílá data pouze pokud příjme libovolný znak, můžeme tímto specifikovat periodu měření. Doba, která toto určuje je parametr readPeriod, přijatá externě, a implementace této doby je uspání vlákna. Tato doba je určena v milisekundách.



Obrázek 2: Program CommBridge

- Poslední vlákno je hlavní smyčka, která vykonává metodu representDataFromDevice. Ta v neustálé smyčce, která je zastavena pokud metoda isStopped konkrétní instance měřidla vrátí hodnotu true, načítá data z měřidla a posílá je na standartní výstup.

3.4 Implementování nového typu měření

Jelikož měkteré typy měřidel mohou vyžadovat unikatní typy nastavovacích parametrů, je pro takové měřidla potřeba upravit program CommBrigde. Celá úprava pak spočívá v přidání názvu nového měřidla do hlavičky třídy CComDeviceFactory a v dále upravit její metodu createDevice tak, aby pomocí tohoto názvu vytvořila konstruktor na nový typ měřidla. Ten pak bude implementován v nově vytvořené třídě, která bude dědit buď ze třídy CComunicationDevice nebo ze tříd, které již na ni mají návaznost jako je třída CSerialComDevice. Třída pro nový typ měřidla pak bude obsahovat atributy reprezentující unikátní parametry daného měřidla a implementaci metod nutných ke komunikaci s měřidlem (send, reciev, connect, stop). Pro přidání nového typu měřidla je také potřeba upravit aplikaci Unisave. Tyto úpravy jsou popsány v kapitole 4.3.2.

4 Aplikace Unisave

Unisave je aplikace napsaná v jazyce Java sloužící k načítání dat z různých typů měřidel a jejich reprezentaci jak v textové tak v grafické podobě. Tato aplikace měla před mými úpravami implementován pouze jeden typ měření a tím bylo měření typu XY, kdy se načítala data z dvojice měřidel. Jelikož úkolem této bakalářské práce bylo načítat data pouze z jednoho měřidla, bylo potřeba doimplementovat nový typ měření. Toto měření navíc implementuje časovou závislost měření, tzn. data se načítají z měřidla ve specifikovaných časových intervalech, zadaných na začátku měření. Tato perioda měření se nedá po dobu měření měnit a je tedy konstantní. Proto jsem tento typ měřeni nazval Periodic. Veškeré úpravy této aplikace byly prováděny výhradně v prostředí Eclipse. Návrh třídních UML diagramů pak bylo vytvořeno aplikací IBM Rational Software Architect, která funguje jako rozšíření IDE Eclipse. Tato kapitola by pak měla popsat následující věci:

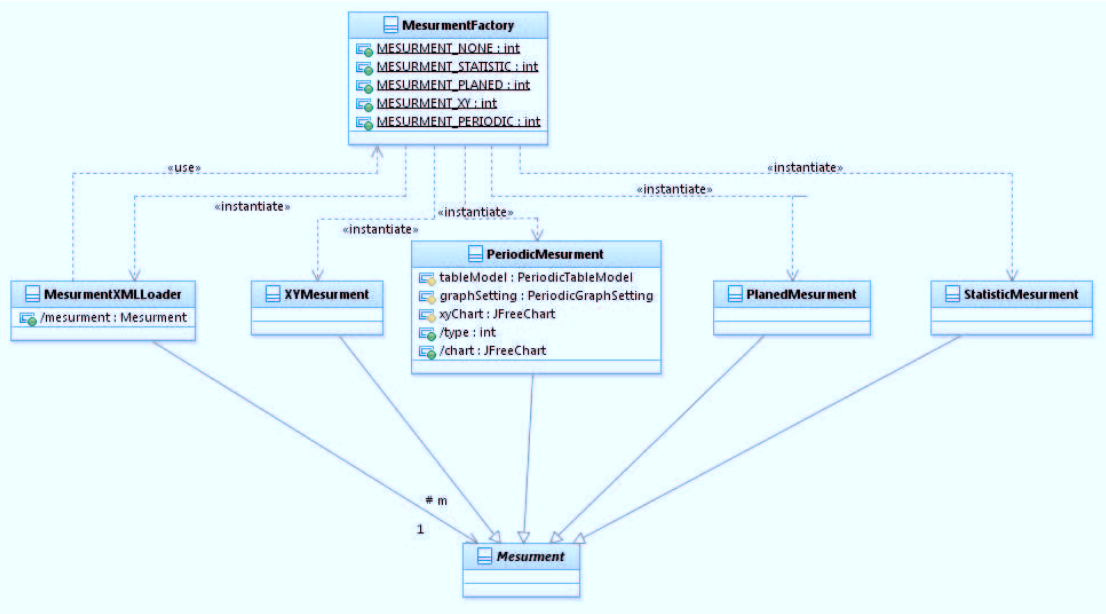
- analýza programu a popis způsobu, jakým se data dostanou z měřidla do reprezentované formy
- jakým způsobem se implementuje nový typ měření
- přidání nového měřida a vytvoření nového typu měřidla

4.1 Analýza aplikace

Jak už bylo zmíněno výše, aplikace Unisave je napsaná vjazyce Java. Struktura aplikace je na kvalitní úrovni, jelikož všechny třídy jsou pěkně rozdělené do balíčku podle své funkce. Balíčky, kterých se bude týkat následná analýza (v hierarchickém pořadí) jsou:

- data
  - value
- device
- grabber
- gui
  - action
  - value
- setting

Mimo tyto balíčky je důležitý ještě balíček resource. Ten obsahuje dva důležité soubory, které nelze upravovat pomocí aplikace Unisave, ale musí se vždy upravit ručně. Oba tyto soubory jsou typu XML. První soubor je mesurementDevice. Ten obsahuje informace o měřidlech jako jejich typ, nastavovací parametry pro měření a jednotky, které každé měřidlo dokaže měřit. U každého typu měřidla se jejich nastavovací parametry liší.

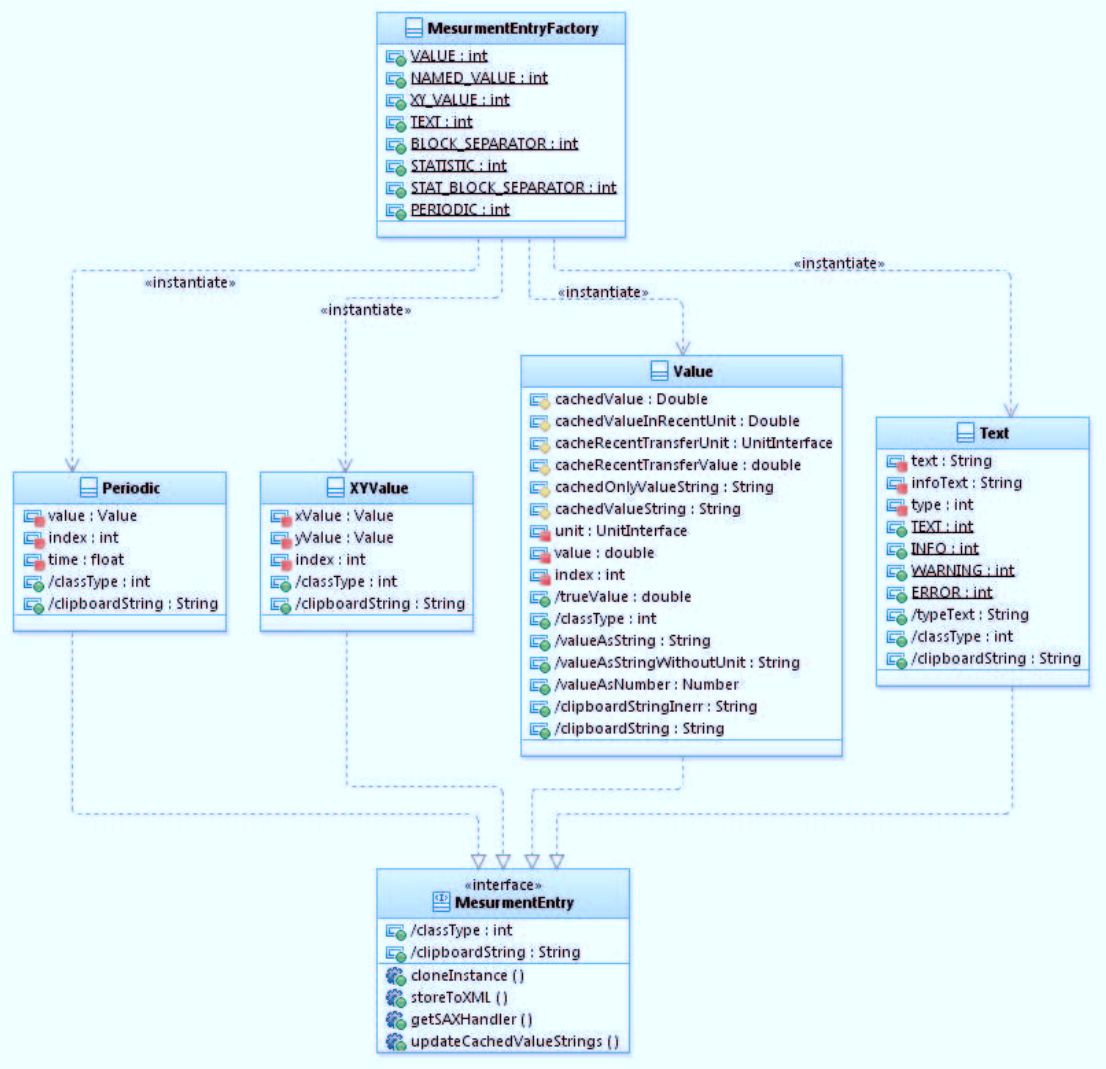


Obrázek 3: Balíček data

Druhým souborem je soubor units, který obsahuje informace o konkrétních jednotkách a možnostech jejich převodu. O modifikaci těchto dvou souborů bude pojednávat kapitola 4.3.1.

4.1.1 Balíček data

Tento balíček, se základní strukturou zobrazenou na obrázku 3 je, dalo by se říci, jakousi kostrou celé datové části. Obsahuje totiž třídy pro jednotlivé typy měření. Tyto třídy tedy v sobě uchovávají informace nutné k reprezentaci načtených a již zpracovaných hodnot. V současnosti jsou v provozuschopném stavu pouze dva typy měření: XYMesurement a PeriodicMesurement. O vytvoření daného typu měření se stará třída MesurementFactory, která podle specifikovaného typu vytvoří instanci měření. Všechny konkrétní měření pouze rozšiřují třídu Mesurment, která obsahuje jednotlivé naměřené data v kolekci datového typu MesurementEntry (více v kapitole 4.1.2). Dále obsahuje uživatelské paramentry, listenery změny v kolekci naměřených dat a základní ukládání a načítání z/do souboru. Konkrétní třídy měření pak implementují hlavně množniny pro reprezentaci dat v tabulce a v grafu, a dědí ze třídy Mesurment. Například třída PeriodicMesurement v sobě obsahuje další třídy PeriodicTableModel, který reprezentuje množinu dat pro vykreslení tabulky, a třídu XYMesurementDataset, která reprezentuje množinu dat pro vykreslení grafu. Dále tyto konkrétní třídy překrývají metody ze třídy Mesurment pro ukládání a načítání do souboru pomocí anotace override.



Obrázek 4: Balíček data.value

4.1.2 Balíček data.value

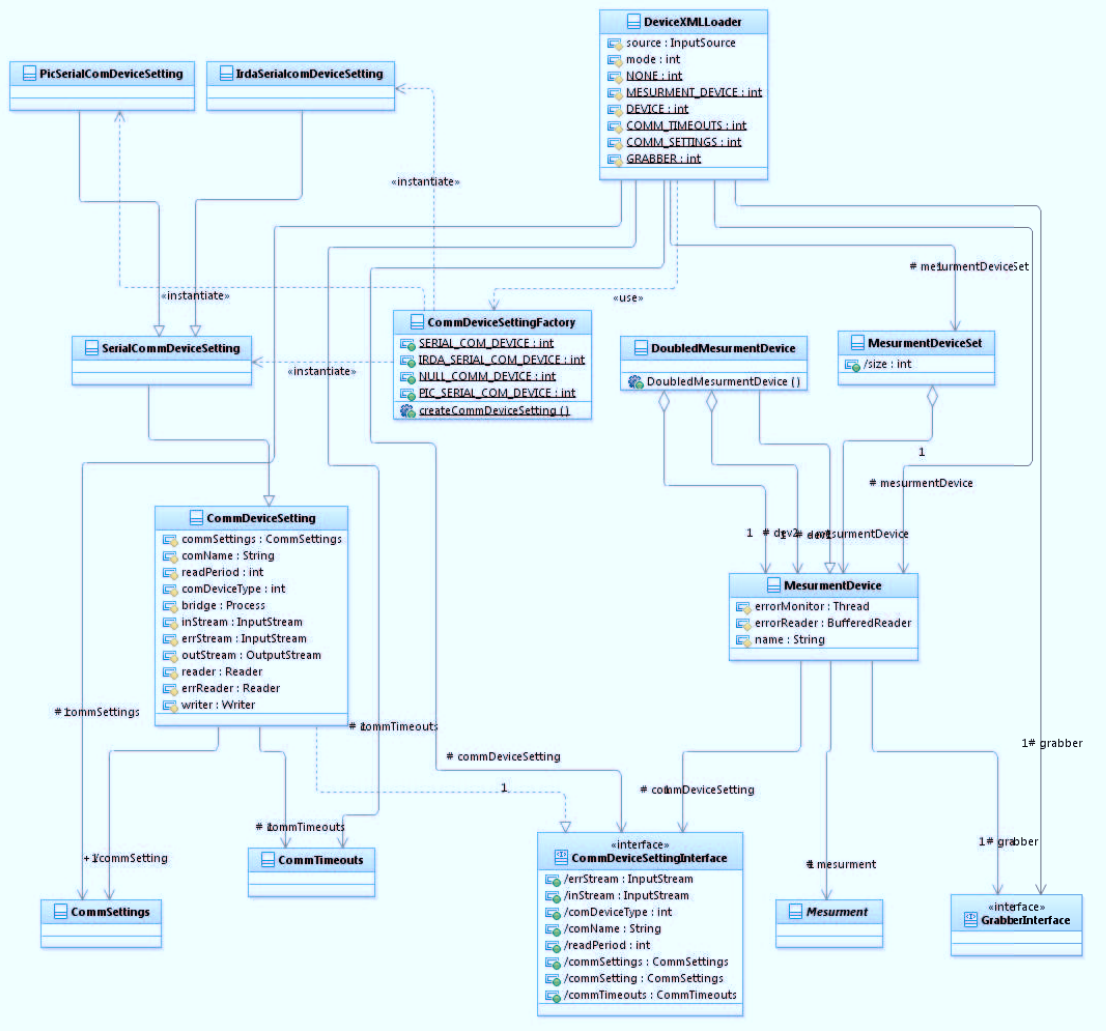
Další částí balíčku data je jeho rozšíření data.value, zobrazený na obrázku 4. To už podle názvu v sobě uchovává třídy, které definují jednotlivé záznamy naměřených dat. Třída MesurementEntryFactory definuje jak enumerátory pro jednotlivé typy měření, tak jejich konstruktory. Jednotlivé typy měření implementují rozhraní MesurmentEntry. Každý typ měření má pak vlastní třídu, která dodefinuje konkrétní parametry, se kterým toto měření pracuje. U třídy XYValue jsou to například dva záznamy s naměřenou hodnotou a u třídy Periodic zase čas, ve kterém měření od počátku nastalo. Mimo tyto třídy pak je třída Value, která také implementuje rozhraní MesurmentEntry, ale naproti třídám jež definují celkový záznam, tato třída definuje pouze naměřenou hodnotu. Důležité parametry, jež obsahuje, jsou:

- číselný údaj naměřené hodnoty datového typu double
- jednotka naměřené hodnoty reprezentována třídou UnitInterface
- index naměřené hodnoty

Mimo tyto parametry obsahuje třída Value také metody pro konverzi jednotek, uložení z/do souboru a další méně důležité metody pro práci s daty.

4.1.3 Balíček device

Jak už název napovídá, tento balíček reprezentuje jak obecné tak konkrétní zařízení, se kterými aplikace Unisave pracuje. Jeho třídní diagram je zobrazen na obrázku 5. Jak už bylo v předchozích balíčcích, tak i tady se konkrétní typ zařízení vytváří v podobné třídě, a to konkrétně CommDeviceSettingFactory. Obecné zařízení je reprezentováno třídou CommDeviceSetting, jenž v sobě uchovává parametry, které jsou ovšem reprezentovány ještě dalšímy třídami. Obsahuje také metody pro spojení s programem CommBridge, kdy tyto metody jak spouštějí tento program, tak také nastavují společné standartní vstupně/-výstupní zařízení, přes které se pak posílají nastavovací parametry. Tyto parametry jsou reprezentovány dvěma třídami. První z nich je třída CommSetting, která přesně kopíruje strukturu DCB popsanou v kapitole 3.2. Druhou je třída CommTimeouts, která podobně jako třída CommSetting kopíruje nastavovací parametry, ale v tomto případě se jedná o strukturu COMMTIMEOUTS popsanou ve stejné kapitole. Obě tyto třídy v sobě obsahují metodu pro parsování parametrů z XML souboru a metodu pro poslání těchto parametrů přes standartní výstup do programu CommBridge. Obecně však pro načítání parametrů slouží třída DeviceXMLLoader. Konkrétní typ zařízení pak rozšiřuje třídu CommDeviceSetting a přidávají vlastní parametry, potřebné pro jednotlivé typy zařízení. Pro tyto parametry pak pomocí anotace override definuje vlastní metodu pro nastavení parametrů, kdy nejprve zavolá metodu z nadřazené třídy pro poslání základních nastavovacích parametrů a následně pošle parametry vlastní. Tyto třídy jsou například SerialCommDeviceSetting či IrdaSerialCommDeviceSetting. Poslední třídou, která spravuje jednotlivé zařízení je třída MesurementDevice. Tato třída je vlastně jediná třída, která je dostupná v ostatních balíčních. Obsahuje v sobě metody pro spuštění a zastavení samotného měření,



Obrázek 5: Balíček device

jednotlivé měření a posluchače na spuštění a zastavení měření, monitoring chyb a vlastní grabber(popsáno v kapitole 4.1.4).

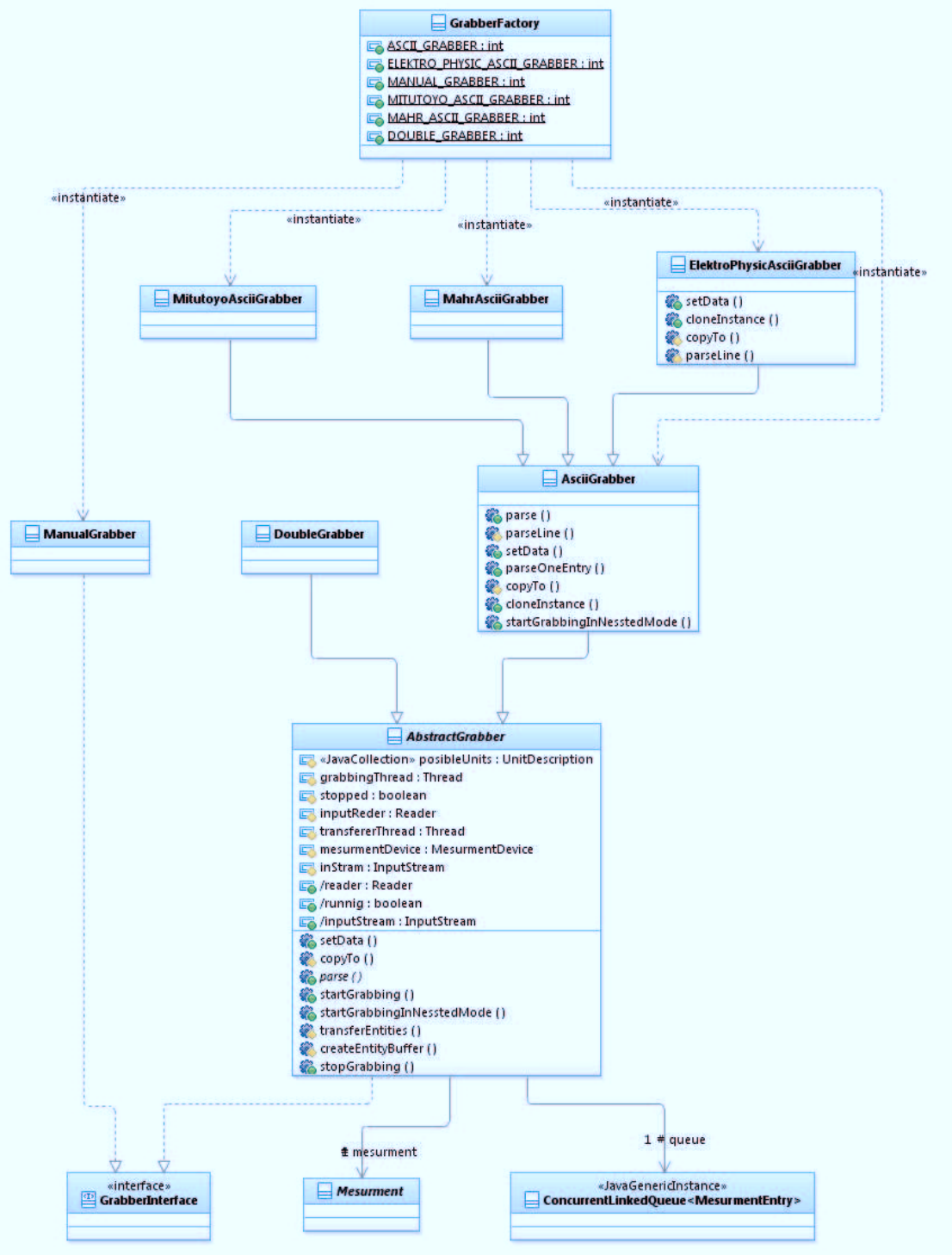
4.1.4 Balíček grabber

Balíček grabber(obrázek 6) v sobě obsahuje třídy, jejichž úkolem není nic jiného, než čtení dat ze standartního vstupu, které posílá program CommBridge. Továrna na vytvoření grabberu je v tomto balíčku GrabberFactory. Základním graberem je zde abstraktní třída AbstractGrabber, která v sobě uchovává několik velice důležitých věcí. První z nich je vstupní stream, který je vlastně standartním vstupem pro načítání hodnot. Další je fronta(queue), která slouží pro přenos naměřených hodnot. A v neposlední řadě je to samotný přenos dat, jež se provádí v samostatném vlákně v metodě transferEntries. Samostatný přenos dat bude popsán v kapitole 4.1.9. Parsování dat se provádí v kontrétních třídách, pro potřeby jednoduchého parsování jednoho řetězce slouží třída AsciiGrabber. Ta připravuje přečtený řetězec do zpracovatelné formy a buď tento řetězec rozparsuje sama nebo, pokud je zvolen jiný typ graberu např. ElektroPhysicAsciiGrabber, pak ho pošle tomuto graberu ke zpracování. Toto zpracování vrací instanci vlastní měrené hodnoty popsané v kapitole 4.1.2.

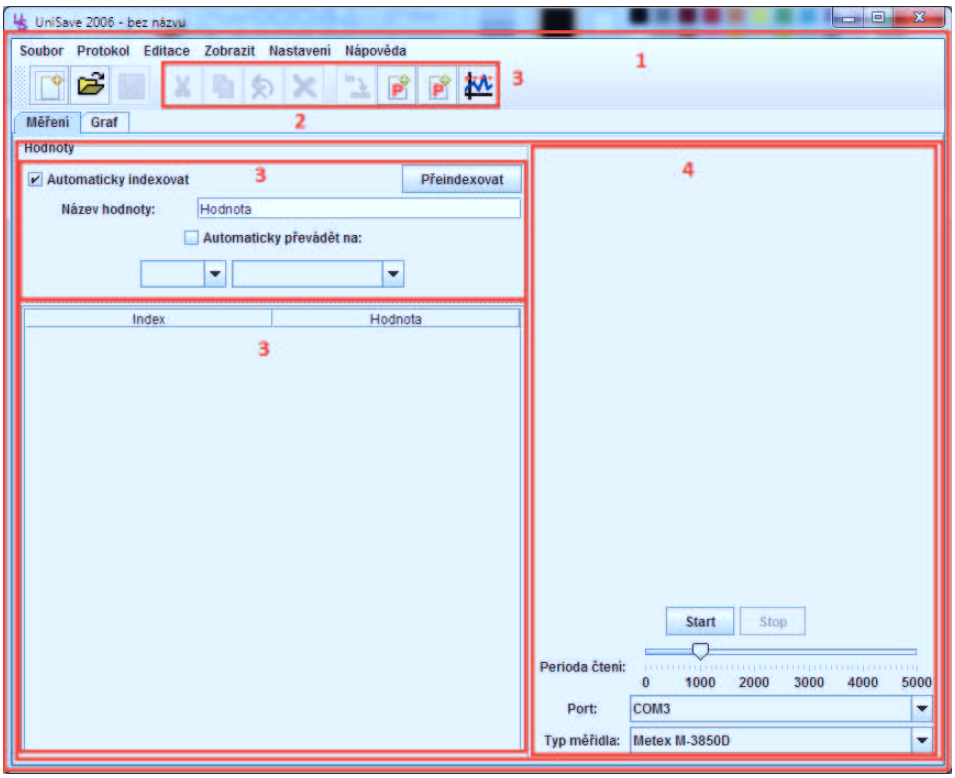
4.1.5 Balíček gui

Jak je z názvu naprosto jasně patné, tento balíček bude obsahovat třídy, jejichž úkolem nebude nic jiného než definovat vzhled a chování grafického prostředí aplikace. Hlavní okno je nadefinováno v třídě MainFrame. Zde se definuje záhlaví tabulky, tzn. celé menu, tlačítková lišta a akce na určité tlačítka(nové měření, uložení, načtení měření) a přidání panelu s měřením do hlavního okna. Panelů s měřením je více a jejich výběr je proveden podle typu měření. Tento výběr je implementován ve třídě MesurmentPanelFactory. Rozhraní, které definuje metody použité v konkrétním panelu měření se nazývá MesurmentPanelInterface. Konktrétní panel s měřením je pak rozdělen na více částí, které jsou:

- akce tlačítek, konstruktor pro konkrétní měření, tabularizace oken pro měření a graf, zobrazení grafu jsou umístěny ve třídě, na kterou se vytvoří konstruktor v MesurmentPanelFactory. Konkrétně se pro např. periodické měření tato třída jmenuje PeriodicMesurementPanel.
- tlačítka související s měřením, panel ovlivňující probíhající měření včetně tabulky, která reprezentuje naměřené hodnoty jsou pro periodické měření umístěny ve třídě PeriodicMesurmentValuePanel.
- tlačítka pro spuštění a zastavení měření, rolovací lišta pro výběr seriového portu, výběr zařízení, z ekterého se načítají hodnoty a perioda čtení jsou umístěny ve třídě GrabberPanel. Pro měření typu XY je však použita třída DoubleGrabberPanel.



Obrázek 6: Balíček grabber



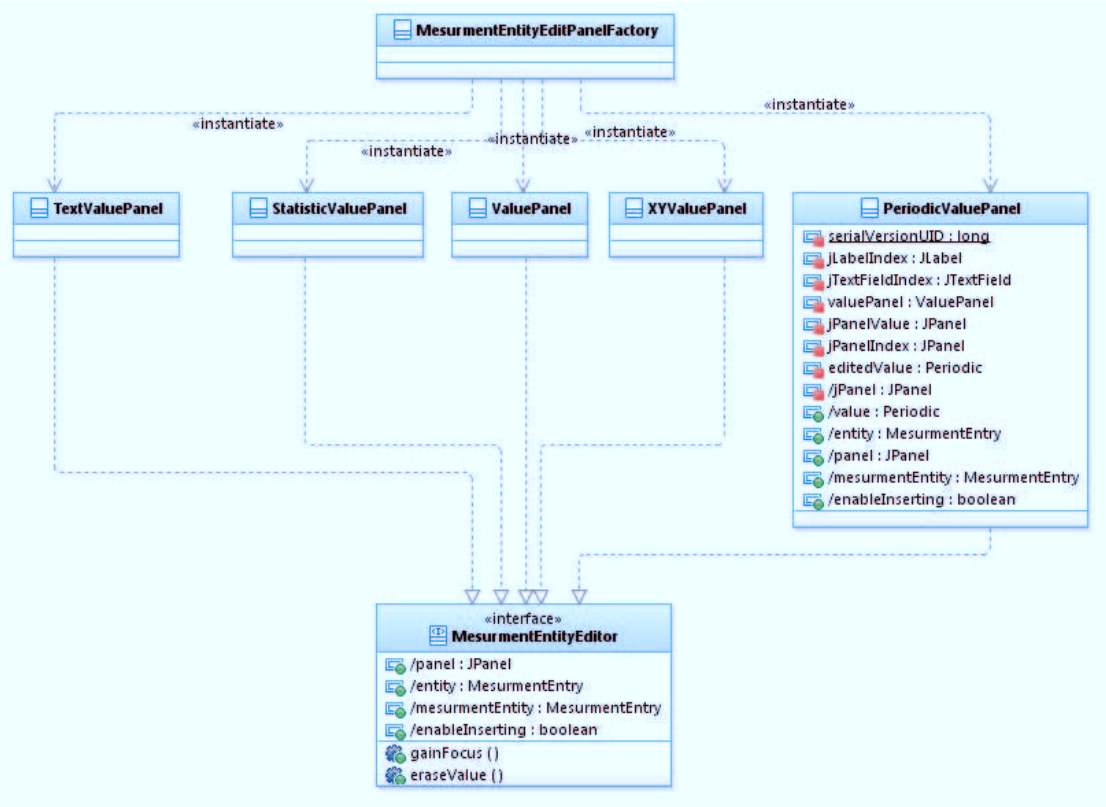
Obrázek 7: Panely aplikace

Úpravy či vytvoření nových tříd, souvisejících blíže s konkrétním typem měření budou vysvětleny v kapitole 4.2.3. Znázornění jednotlivých prvků grafického rozhraní pro měření typu Periodic najdete na obrázku č. 7. Čísla na obrázku odpovídají těmto třídám:

- č. 1 - třída MainFrame
- č. 2 - třída PeriodicMesurmentPanel
- č. 3 - třída PeriodicMesurmentValuePanel
- č. 4 - třída GrabberPanel

4.1.6 Balíček gui.action

Tento balíček obsahuje třídy, rozšiřující třídu AbstractAction. Každá se zde umístěných tříd představuje jedno tlačítko či položku v menu, které se předává pomocí konkrétní třídy v tomto balíčku ikona, její klávesová zkratka, popis a jejich aktivnost(zda je tlačítko možné stisknout při inicializaci). Také je u každé třídy prázdná metoda actionPerformed, která je pak přepsána při vykonání kódu ve třídách z balíčku gui.



Obrázek 8: Balíček gui.value

4.1.7 Balíček gui.value

Zde se definuje sada tříd, sloužících pro editaci hodnot v tabulce či vložení hodnot do tabulky ručně. Opět je tu třída MesurmentEntityEditPanelFactory, která definuje, které třídy se použijí pro zobrazení editačního dialogu. Konkrétní třídy pak obsahují panely, které umožňují editaci hodnot určených právě pro konkrétních typů měření. Například třída XYValuePanel implementuje panely pro editaci a zobrazení dvou hodnot a třída PeriodicValuePanel umožňuje naproti tomu editaci jedné hodnoty a času, kdy byla hodnota změřena. Základní struktura tohoto balíčku je zobrazena na obrázku 8.

4.1.8 Balíček settings

Opět je z názvu jasné užití tohoto balíčku. Slouží pro uchování nastavitelných údajů pro celou aplikaci a jejich načítání či uložení do souboru. Z tříd zde obsažených je nutné pro pochopení funkčnosti znát pouze dvě. Třída GlobalSetting obsahuje inicializace načítání XML souborů mesurementDevice, units, souboru CommBridge.exe a další. Naproti tomu třída UserSetting obsahuje všechny nastavitelné parametry uvnitř aplikace jako názvy používých portů, formát zobrazovaného řetězce hodnot, poslední použité zařízení a pod.

Z toho důvodu musí implementovat mimo čtení ze souboru také zápis do tohoto souboru. Všechny tyto parametry se ukládají do souboru typu XML umístěného v kořenovém adresáři domovské složky uživatele pod názvem UniSave2006_UserSetting.

4.1.9 Cesta reprezentace dat

Jelikož data od vlastního načtení prochází mnoha třídami, bude lepší si celou tuto cestu od načtení dat až po jejich textovou a grafickou reprezentaci popsat (některé části lze vidět i na obrázku 6). Pro popis budeme uvažovat použití periodického měření a využití grabberu typu ElektroPhysicAsciiGrabber, které se oba používají pro načítání hodnot pro měřidlo METEX. Pokud neuvažujeme nutné nastavení měřidla a jeho spuštění v programu CommBridge, první skutečné načtení dat v aplikaci bude ve třídě AsiiiGrabber, konkrétně v metodě parse. Tato metoda ve smyčce neustále načítá data ze vstupního streamu do řetězce, který poté upraví do StringBuilderu tak, aby jedna instance představovala pouze jeden naměřený řetězec. Ten potom přechází do třídy ElektroPhysicAsciiGrabber, kde metodou parseLine rozkouskuje tento řetězec na části, z nichž nejdůležitější je číselný údaj a značka jednotky. Dále se vytvoří instance třídy Periodic pro konkrétní záznam, který se pošle zpět do metody parse, ovšem jako typ MesurmentEntry. Tam se k němu, pokud je vytvořeno indexování, nastaví index a časový údaj. Následně se tento záznam přidá do sdílené fronty. Tato fronta se zpracovává v odlišném vlákně a to konkrétně jako metoda transferEntries v abstraktní třídě AbstractGrabber. V této metodě se vytáhnou všechny data, která obsahuje a převedou se na kolekci datových typů MesurmentEntries, která se následně předá instanci třídy Mesurment a vymaže všechny záznamy. Tím je zajištěno předání pouze nových hodnot. Toto vlákno je následně uspáno, tzn. vykonává se pouze každých 600ms. Třída Mesurment obsahuje vlastní kolekci typů MesurmentEntry, do které se po vyvolání metody addEntries předají nově naměřené hodnoty. Tato metoda pak vyvolá událost, že byly přidány nové záznamy a všichni posluchači, jako je množina pro vykreslení dat v tabulce či grafu na tuto událost zareagují tím, že tyto data přidají do vlastních záznamů čimž je zobrazí.

4.2 Implementace nového typu měření

Z důvodů implementace pouze jednoto typu měření v době, kdy byla aplikace dodána, bylo nutné naimplementovat nový typ měření a poskytnout popis, jakým se implementace nového typu měření provádí. Jak již bylo zmíněno v úvodu kapitoly, nové měření bylo nazváno Periodic. Pro implementaci nového měření bylo třeba upravit několik tříd a hlavně vytvořit třídy zcela nové. Jelikož změny byly potřeba ve více balíčcích, následující kapitoly budou popisovat změny pro vytváření měření v nových balíčcích.

4.2.1 Implementace nového měření v balíčku data

První změnou, nutnou pro implementaci nového typu měření je potřeba provést ve třídě MesurmentFactory. Zde je potřeba vytvořit enumerátor pro nové měření a upravit metoru

createMesurment tak, aby instance třídy Mesurment použila konstruktor třídy nově vytvořeného měření. Pro typ Periodic byla vytvořena nová třída PeriodicMesurment. Další třídy, které jsou ve třídě PeriodicMesurment použité jsou následující:

- PeriodicGraphSetting - obsahuje dodatečné údaje pro tvorbu grafu a metody zajišťující uložení či načtení techto hodnot
- PeriodicGraphSettingListener - rozšiřuje GraphSettingListener a umožňuje přidání reakcí na nové typy událostí. V současném stavu není využita žadná nová událost.
- PeriodicMesurmentSettingListener - umožňuje přidání reakcí na nové typy událostí v konkrétním měření

Samotná třída PeriodicMesurment je pouze rozšířením třídy Mesurment. V základu obsahuje pouze dvě nejdůležitější součásti, které je třeba implementovat pro vytvoření nového měření. První je model pro vytvoření tabulky. Ten je reprezentován třídou PeriodicTableModel uvnitř třídy PeriodicMesurment, která jako parametr uchovává instanci této třídy. Tento model rozšiřuje AbstractTableModel a implementuje rozhraní na posluchače MesurmentEntryListener a PeriodicMesurmentSettingListener. Jelikož je nastaven jako posluchač na události implentovanými v těchto třídách, musí reagovat hlavně na přidání nových záznamů. Rovněž přeposílá tyto události dále do gui. Druhou součástí je graf společně s množinou dat nutnou pro jeho vykreslení. Pro tuto množinu platí stejné zásady, jako pro množinu dat v tabulce. Poslední věcí nutnou k implementaci jsou třídy nutné k uložení a načtení parametrů, lež náleží této třídě.

4.2.2 Implementace nového měření v balíčku data.value

Zde platí totéž co u každého balíčku. Je tedy nejprve třeba upravit třídu MesurmentEntryFactory, tedy přidat enumerátor na vlastní typ měřené hodnoty a poté přidat i nový konstruktor do metody getMesurmentEntry a nejlépe i přidat popisek do metody getName, která vrací popis daného typu value. Dále je nutné vytvořit novou třídu, která naměřené hodnoty bude reprezentovat. Jelikož nově vytvořená třída musí implementovat rozhraní MesurmentEntry je jasné, které metody bude potřeba doimplementovat (ty, které jsou obsaženy v tomto rozhraní).

4.2.3 Implementace nového měření v balíčku gui

Úprava balíčku gui je lehce složitější než ostatní balíčky a to z důvodu, že pro nový typ měření se může upravit velice hodně nebo také se nemusí upravovat vůbec nic. V zásadě jde pouze o úpravu prvků v oblasti měření, což zahrnuje obecné třídy MesurmentPanel, MesurmentValuePanel, UnitConversionDialog. Dále je možné přidat třídy typu GrabberPanel, který by byl závislý na jednotlivém typu měření. V kapitole 4.1.5 je popsáno, co která třída implementuje a tedy co každá nově vytvořená třída musí obsahovat, aby byla zaručena správná funkčnost programu. Pro tvorbu nového panelu pro měření je třeba

jako v předchozích případech vytvořit konstruktor pro toto měření ve třídě MesurmentPanelFactory. Pro jakékoliv úpravy je nutno dobrých znalostí pro tvorbu grafických prvků v jazyce java. Úprava panelu nastavení je blíže popsána v kapitole 4.4.1.

4.2.4 Implementace nového měření v balíčku gui.value

Protože každý nový typ měření by měl v tabulce jiné hodnoty parametrů, je třeba upravit tento balíček, kdy ve třídě MesurmentEntityEditPanelFactory vytvoříme konstruktor, vyvolaný podle enumerátoru vytvořeného podle popisu v kapitole 4.2.2. Tento konstruktor bude přiřazen na nově vytvořenou třídu nutnou pro editaci konkrétních hodnot v novém typu měření. Pokud bude vytvořena nová třída bude nutné panely upravit tak, aby v panelu bylo možno upravit všechny veličiny, které se pro dané měření využívají.

4.3 Přidání nového měřidla a implementace přidání nového typu měřidla

4.3.1 Přidání nového měřidla

Jelikož aplikace neobsahuje žádnou možnost, jak přidat nové měřidlo v rámci grafického prostředí, je nutné pro každé měřidlo vložit záznam do souboru mesurmentDevice.xml, uloženého v adresáři resource. Pro vložení nového měřidla je nutné znát následující věci:

- Název měřidla - název použitý při zobrazení v aplikaci
- Typ měřidla - použitý typ měřidla, seznam implementovaných měřidel a číslo typu je zobrazen v tabulce 3
- Parametry sloužící konkrétnímu typu měřidla
- Nastavovací parametry pro sériovou linku
- Použitý typ grabberu - názvy a čísla typu zobrazeny v tabulce 4
- Parametry pro grabbování: - zapnutí autoindexace, EndChar, Offset, Truncate
  - zapnutí autoindexace
  - valueEndChar - ascii číslo znaku, kterým je zakončen každý přenesený řetězec dat
  - valueOffset - počet znaků od začátku, jež nebudou použity při parsování přijatého řetězce
  - valueTruncate - počet znaků od konce, jež nebudou použity při parsování přijatého řetězce
- Možné jednotky měřené měřidlem. Aktuálně implementované jednotky s daným indexem lze nalézt v tabulce 2

Index	Název jednotky	Zkratka jednotky
1	Procento	%
3	Metr	m
4	Mikropalec	microinch
5	Milipalec	mil
6	Palec	in
7	Stopa	ft
8	Yard	yd
9	Míle	mi
10	Newton	N
11	Volt	V
12	Amper	A
13	Odpor	Ohm

Tabulka 2: Tabulka implementovaných jednotek

Číslo typu měřidla	Název typu
1	Sériové zařízení
2	Infračervené zařízení
3	Zařízení typu null
4	Mikroprocesorové zařízení

Tabulka 3: Tabulka typů měřidel

Při znalosti těchto parametů je třeba v dané struktuře vložit záznam. Strukturu je možno zjistit z již vložených měřidel nebo rozkódovat přes načítání těchto parametrů v balíčku device.

Dalším souborem, který je případně nutné editovat je soubor units.xml. Ten v sobě uchovává informace o užívaných jednotkách. Je tedy nutné ho editovat pouze v případě, že měřené jednotky nejsou zatím vytvořeny. Pro přidání nové jednotky je potřebná znalost následujících informací:

- Název jednotky
- Běžně užívaná zkratka jednoty
- Zda jednotka může užívat prefixy(mili, kilo, giga atd.)
- Zda se jednotka může převést na jednotku jinou. To je dáno následujícím vzorcem:

$$To = \frac{(From + offset) \cdot multiplier}{divider} + offset2$$

Číslo typu grabberu	Název grabber
1	Ascii grabber
2	Elektro-physic ascii grabber
3	Manual grabber
4	Mitutoyo ascii grabber
5	Mahr ascii grabber
6	Double grabber

Tabulka 4: Tabulka typů grabberů

4.3.2 Implementace přidání nového typu měřidla

Pokud budeme z jakéhokoliv důvodu potřebovat implementovat nový typ měřidla, musíme je nejprve se všeho ujistit, že nový typ měřidla je již implementovaný v programu CommBridge, jelikož spolu s aplikací Unisave blízce komunikují. Pro nový typ měřidla je potřeba provést úpravy v balíčku device. Tam se nachází třída CommDeviceSettingFactory, která definuje konstruktory a enumerátory jednotlivých typů zařízení. Zde je potřeba přidat nový typ zařízení a přidat konstruktor na tento nový typ zařízení. Nová třída, reprezenující nový typ zařízení, musí dědit buď ze třídy CommDeviceSetting nebo ze třídy, která z ní již dědí. V nové třídě je nutné dodefinovat nové parametry, specifické pro daný typ zařízení a napsat implementaci pro přečtení z xml souboru(metoda setData) a implementaci pro zápis do programu CommBridge(metoda sendCommSetting).

4.4 Další úpravy aplikace Unisave

4.4.1 Změny provedené v dialogovém okně nastavení

Původně mělo okno nastavení pouze pevnou sktrukturu, která byla pro další rožšíření již nevyhovující. Proto bylo přistoupeno ke změně tohoto okna na tabularizovatelné (rozděleno na záložky), kdy v první záložce zůstaly prvky z původního okna a v dalších záložkách bude možné implementovat dodatečné nastavovací prvky. V současném stavu je naimplementována záložka pro nastavení periodického měření, kde je použit pouze jeden prvek a to zaškrkovací políčko, které povolí či zakáže dialogové okno s varováním v době, kdy je již naměřeno jedno měření a uživatel si přeje spustit nové tlačítkem start. Dalším novým prvkem je změna tlačítka výchozí, kdy v původním stavu toto tlačítko pouze prepsalo seznam použitelných portů na porty COM1 - COM4. Nově toto tlačítko spustí externí program EnumSer.exe [4], který načte aktuální porty užité v počítači. Tento program musel být modifikován, jelikož v původní podobě obsahoval více metod jak zjistit aktuální porty v počítači a u každé metody vypisoval čas, za který bylo touto metodou výčet portů zjištěn. Vybrán byl proto ten nejrychlejší a výpis z tohoto programu byl upraven tak, aby vypisoval na standartní výstup pouze názvy portů.

4.4.2 Ostatní změny v aplikaci

Další změny v aplikaci jsou už pouze minoritní. Je to úprava třídy s nastavením uživatelských prarametrů a jejich ukládání i načítání ze souboru. Dále bylo doimplementováno načítání hodnot podle časové závislosti nezávisle na měřidlu a úprava třídy pro parsování dat, aby bylo možno načítat specifická data z měřidel METEX a Protek. Poslední úpravou je přidání nového typu měřidla pro komunikaci, která nesouvisí s touto bakalářskou prací, ale pro pochopení aplikace není od věci ji zmínit. Zařízení je v tomto případě mikroprocesor PIC16F877A od firmy Microchip, kdy tomuto zařízení je potřeba předat jako parametr číslo měřeného kanálu, jež je uložen v souboru mesurmentDevice. Z toho důvodu bylo potřeba doimplementovat tento typ zařízení. Postup k této implementaci je v kapitole 4.3.2.

5 Závěr

Jelikož hlavním cílem této bakalářské práce bylo rozšíření aplikace Unisave o měřidlo elektrických veličin značky METEX, dá se říci, že cíl byl splněn. Aplikace Unisave byla nejen rozšířena o možnost načítání z tohoto měřidla. Dále bylo přidáno měřidlo značky ProTek a v textu bakalářské práce byl popsán obecný postup pro přidávání dalších měřidel i typů měření.

Co by se dalo dále podoktnout je zkušenost vzniklá při zpracovávání tématu této práce, kdy manuál dodávaný k měřidlu METEX obsahoval informace, jež neodpovídaly skutečnosti. Z toho důvodu musela být provedena analýza komunikace mezi mezi tímto měřidlem a programem, který k němu byl dodáván.

Dále bylo popsáno mnoho užitečných informací, jako popis nejdůležitějších parametrů sloužících ke spojení měřidla skze jazyk C++, a také byla popsána celková komunikace v tomto jazyce s přístrojem. V poslední řadě bylo upravení aplikace Unisave, které zabralo nejvíce času, jelikož je tato aplikace ve svém současném stavu lehce složitější na pochopení, ale doufám, že tato práce poslouží pro snažší pochopení této aplikace včetně následného doimplementování.

Matěj Děcký

6 Reference

[1] Matoušek David *USB prakticky s obvody FTDI* Nakladatelství BEN, 2003.

[2] Msdn.microsoft.com [online]. 2011-10-03 [cit. 2011-05-01]. DCB Structure. Dostupné z WWW: http://msdn.microsoft.com/en-us/library/aa363214%28v=vs.85%29.aspx.

[3] Msdn.microsoft.com [online]. 2011-10-03 [cit. 2011-05-01]. COMMTIMEOUTS Structure. Dostupné z WWW: http://msdn.microsoft.com/en-us/library/aa363190%28v=vs.85%29.aspx.

[4] Naughter.com [online]. 2011 [cit. 2011-05-01]. EnumSerialPorts. Dostupné z WWW: http://www.naughter.com/enumser.html.

[5] Technet.microsoft.com [online]. 2006 [cit. 2011-05-01]. Portmon for Windows v3.02 Dostupné z WWW: http://technet.microsoft.com/en-us/sysinternals/bb896644.